



SQL



History

- IBM developed the original version of SQL in the early 1970s
 - Originally called Sequel
- Its name has changed to SQL (Structured Query Language)
- SQL has become the standard relational database language
- Published as a standard SQL in 1986 by
 - the American National Standards Institute (ANSI) and
 - the International Organization for Standardization (ISO)



SQL Standards

- First SQL standard
 - SQL-86
- ANSI published an extended standard for SQL
 - SQL-89,in 1989
 - Now called – SQL1
- The next version of the standard was
 - SQL-92 standard
 - SQL2
- Then followed by SQL:1999
 - SQL3



SQL – Sub-Languages

- SQL language can be used to:
 - Define the structure of the data
 - Modify data in the database, and
 - Specify security constraints
- Data Definition (DDL)
 - Commands to create the structure of schemas and tables.
 - Includes those commands concerned with declaring entity, referential and domain integrity (Data integrity)
- Data Manipulation (DML)
 - Commands for retrieving, updating and inserting data in tables
- Data Control (DCL)
 - Commands concerned with controlling access to the database



DDL Sublanguage

- DDL Sublanguage consists of commands that create, change and remove database objects from the database
- Examples of DDL Statements are:
 - CREATE DATABASE
 - ALTER DATABASE
 - DROP DATABASE
 - CREATE TABLE
 - ALTER TABLE
 - DROP TABLE
 - CREATE CONSTRAINT
 - ALTER CONSTRAINT
 - DROP CONSTRAINT



Create Database

```
CREATE DATABASE database_name
[ ON
    [ PRIMARY ]
    [ <filespec> [ ,...n ]

    [ , <filegroup> [ ,...n ] ]

    [ LOG ON { <filespec> [ ,...n ] } ]
]
```



Create Database - Example

```
CREATE DATABASE ABC_Supermarket_DB
ON PRIMARY
    ( NAME=' ABC_Supermarket_DB_Primary',
      FILENAME= 'D:\ Data\ ABC_Supermarket_DB _Prm.mdf',
      SIZE=4MB,
      MAXSIZE=10MB,
      FILEGROWTH=1MB)
FILEGROUP ABC_Supermarket_FG1
    ( NAME = 'MyDB_FG1_Dat1',
      FILENAME = ' D:\ Data \ ABC_Supermarket_DB_FG1_1.ndf',
      SIZE = 1MB,
      MAXSIZE=10MB,
      FILEGROWTH=1MB)
LOG ON
    ( NAME=' ABC_Supermarket_DB_log',
      FILENAME = ' D:\ Data \ ABC_Supermarket_DB.ldf',
      SIZE=1MB,
      MAXSIZE=10MB,
      FILEGROWTH=1MB)
```



Create Table

- To create a table using SQL you specify:
 - ☐ Name of the table
 - ☐ Name of each of the columns in the table
 - ☐ Data type of each column
 - ☐ Maximum length of each column
 - ☐ Some additional constraints

```
CREATE TABLE <table name>  
(    <column name> <data type> [<length>]  
      , <column name> <data type> [<length>]  
      , .. .. .  
)
```



Create Table - Syntax

```
CREATE TABLE [ database_name.[owner].] table_name
(   <column name> <data type>
        [      [DEFAULT <constant expression>]
            | [IDENTITY [(seed, increment)] ] ]
        [NULL|NOT NULL]
        [<column constraints>]
    ]
    | [ column_name AS computed_column_expression]

    | [ <table_constraint>]
    [ ,...n]
)
[ON {<filegroup>|DEFAULT}]
[TEXTIMAGE_ON {<filegroup>|DEFAULT}]
```



Table and Column Names

- Common practices in naming database objects:
 - For each word in the name, capitalize the first letter and use lowercase for the remaining letters
 - Use of Underscores
- Keep the name short, but make it long enough to be descriptive
- Should also be descriptive to others!
- Whatever your choice – make it your standard, be Consistent
- In SQL Server a **fully qualified object name** has 4 parts
 - ServerName.DatabaseName.OwnerName.ObjectName
 - Also referred as a **4-Part Name**



Data Types

- A data type is an attribute that specifies the type of data that the column can hold
- Data types act in part as a definition for domains
 - Every data value within a column must be of the same type
 - there is no default data type
- Data type Categories:
 - Numeric Data Types
 - Character / String Data Types
 - Date Data Types
 - Binary Data Types
 - Special Data Types



Data Types (cont.)

- There may be a difference in the standard SQL data types and data types implemented by different vendors
- Examples of standard data types

```
CREATE TABLE Sales(  
    [SalesID] INTEGER IDENTITY(1,1) NOT NULL  
    , [CustomerID] CHARACTER(3) NOT NULL  
    , [ItemID] CHARACTER VARYING(3) NOT NULL  
)
```

- Products that are **compliant** with standards will support the standard data types

Data Types - SQL Server

Exact numerics	
bigint	numeric
bit	smallint
decimal	smallmoney
int	tinyint
money	
Approximate numerics	
float	real
Date and time	
date	datetimeoffset
datetime2	smalldatetime
datetime	time
Character strings	
char	varchar
text	

Unicode character strings	
nchar	nvarchar
ntext	
Binary strings	
binary	varbinary
image	
Other data types	
cursor	rowversion
hierarchyid	uniqueidentifier
sql_variant	xml
Spatial Geometry Types	Spatial Geography Types
table	



Default

- This is the value you want to be used for any rows that are inserted without a user-supplied value for this particular column
 - If no value is supplied, the default will always be used
- Defaults are only used in INSERT statements
- The default value should be of the same data type as the column
- Defaults can be anything that evaluates to a constant



DEFAULT Constraints

- Defaults can specified using:

- ☐ a constant,
- ☐ built-in function, or
- ☐ mathematical expression

- Example

```
CREATE TABLE Employee
(
    EmployeeID int IDENTITY NOT NULL PRIMARY KEY
    , IsActive BIT DEFAULT 1
    , preference varchar(30) DEFAULT 'UNKNOWN'
    , dob dateTime DEFAULT GetDate()
)
```



Identity Column

- A sequenced number is generated when you add a row
- The number starts from the SEED value, and increases or decreases by the INCREMENT
- An identity column must be numeric
 - it is almost always used with an integer or BIGINT data type
 - Can it be decimal?
- Cannot be used in conjunction with a default constraint
- IDENTITY is usually used as the PRIMARY KEY column
 - **BUT**, just because you have an IDENTITY column doesn't mean that the value is unique



Identity Column (cont.)

- To insert explicit values into an IDENTITY column
 - Use SET IDENTITY_INSERT ON statement
 - This allows you to turn off the identity process for inserts from the current connection
- At any time, only one table in a session can have the IDENTITY_INSERT property set to ON
- When SET IDENTITY_INSERT TableName OFF is issued, the new inserted value is used as the current identity value
- Example

```
SET IDENTITY_INSERT Sales ON
INSERT Sales( SalesID, CustomerID, ItemID) VALUES ( 5, 103, 1192)
```



NULL / NOT NULL

- States whether the column in question accepts NULL values or not.
- The default, when you first install SQL Server, is to set a column to **NULL**
 - i.e. if you don't specify nullability
- It is recommended that you explicitly state the NULL option for every column in every table
 - There are a large number of different settings that can affect what the system uses for a default for the nullability of a column.



Computed Columns

- A column that doesn't have any data of its own
 - Its value is derived on the fly from other columns in the table
- syntax:
 <column name> AS <computed column expression>

- Example

```
CREATE TABLE Sales(  
    SalesID int PRIMARY KEY  
    , ItemID int  
    , UnitPrice decimal(5,2)  
    , Quantity int  
    , Amount AS UnitPrice * Quantity  
)
```



Column Constraints

- Rules that you place on individual columns
 - about the data that can be inserted into that column
- Can be defined as
 - a Column-Level Constraint definition, or
 - a Table-Level Constraint definition
- Table-Level constraints may be based on multiple columns
 - Constraints that involve more than one column cannot be defined at the column-level
- Example of constraints that can be defined at the table-level
 - PRIMARY and FOREIGN KEY constraints
 - CHECK constraints



ON and TEXTIMAGE_ON

■ ON

- The ON clause in a table definition is a way of specifically stating on which filegroup you want the table located.
- If ON is not used, the table will be placed on whatever the default filegroup

■ TEXTIMAGE_ON

- Lets you move a very specific part of the table to a different filegroup.
- only valid if your table definition has text, ntext, or image column(s) in it.
- Only the BLOB information is moved into the separate filegroup
 - the rest of the table stays either on the default filegroup or with the filegroup chosen in the ON clause.



Info on Tables and Constraints

- Use the following system stored procedures to get Information on Tables and Constraints
 - EXEC sp_help <table name>
 - EXEC sp_helpconstraint <table name>